SkiService API

Modul 165 - NoSQL-Datenbanken einsetzen

By Robin Ruf

Table of contents

01

Informieren

- Projektumfang ermitteln
- Anforderungsanalyse

2.1 Zusammenfassung der Anforderungen

Nr.	Beschreibung
A1	Datenbasis aus relationaler Datenbank vollständig nach NoSQL migriert
A2	Benutzerkonzept mit min. 2 Benutzeranmeldungen mit verschiedenen Berechtigungsstufen implementiert.
A3	Für die Web-API Applikation muss ein eigener Datenbankbenutzerzugang mit eingeschränkter Berechtigung (DML) zur Verfügung gestellt werden
A4	Schema für Datenkonsistenz implementiert
A5	Datenbank Indexe für schnelle Ausführung von Suchabfragen implementiert
A6	Backup und Restore Möglichkeiten umgesetzt (Skript-Dateien)
A7	Vollständige Datenbankmigration mittels Skript-Dateien realisiert
A8	Das Web-API Projekt (CRUD) komplett auf NoSQL Datenbanksystem migriert
A9	Datenmodell vollständig dokumentiert, inkl. Grafik zum Datenmodell
A10	Einfaches Testprojekt in Postman erstellt.
A11	Das Softwareprojekt ist über ein Git-Repository zu verwalten.
A12	Ganzes Projektmanagement muss nach IPERKA dokumentiert sein

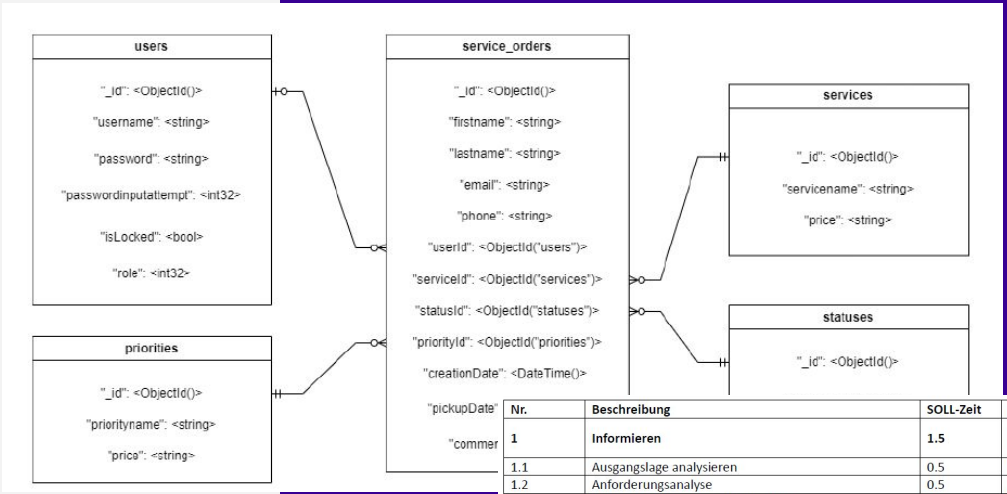
3 Zusätzliche Anforderungen

Zusatzpunkte für optionale Erweiterungen. Zur Erreichung der max. Punktzahl müssen zwei optionale Anforderungen umgesetzt werden. Es werden nur zwei zusätzliche Anforderungen bewertet.

Nr.	Beschreibung
AO1	Automatisiertes Backup-Konzept durchgeführt u. implementiert.
AO2	Komplexe Schema Validierungen umgesetzt (Referenzen, enum, min, max. usw)
AO3	Datenmigrationsskripte zu den RDBMS nach NoSQL realisiert
AO4	Komplexes Datenmodell mit mehr als 6 Grundtypen (Collection / Labels) implementiert
AO5	Komplexe statistische Auswertungsabfragen realisiert

02
Planen

- Zeitplan
- Datenbankmodell
  - Collections
  - Relations
- API-Endpoints



Nr.	Beschreibung	SOLL-Zeit	IST-Zeit
1	**Informieren**	**1.5**	**1.5**
1.1	Ausgangslage analysieren	0.5	0.5
1.2	Anforderungsanalyse	0.5	0.5
1.3	Benötigte Plugins in Erfahrung bringen	0.5	0..5
2	**Planen**	**3.5**	**4.0**
2.1	Erstellen eines Zeitplans	0.5	0.5
2.2	Datenbankdesign entwerfen	1.0	1.5
2.3	API Endpoints planen	1	1
3	**Entscheiden**	**0.5**	**0.5**
3.1	Wo liegt der Fokus?	0.5	0.5
4	**Realisieren**	**32.0**	
4.1	Plugins installieren und konfigurieren	3.0	2.5
4.2	API/Backend Projekt realisieren	5	5
4.3	Datenmigration	5	7
4.4	Backup- und Restore-Skripte schreiben	4	
4.5	Initialisierungsskript schreiben (auto. DB-Creation etc)	8	
4.6	Postman-Testprojekt erstellen	2	3.5
5	**Kontrollieren**	**2.0**	**2.0**
5.1	Postman-Testprojekt ausführen	0.5	0.5
5.2	API-Endpoints mit Swagger testen	1.5	1.5
6	**Auswerten**	**5.25**	**5.75**
6.1	Dokumentation nach IPERKA fertigstellen	3.5	4.0
6.2	Präsentation (PDF) erstellen	1.0	1.0
6.3	GitHub Repo vervollständigen	0.5	0.5
6.4	Abgabe des GitHub-Repo-Links	0.25	0.25
Gesamt		**37.25**	

03

Entscheiden

- Wo liegt mein Fokus?
  - Zeitmangel...

04

Realisieren

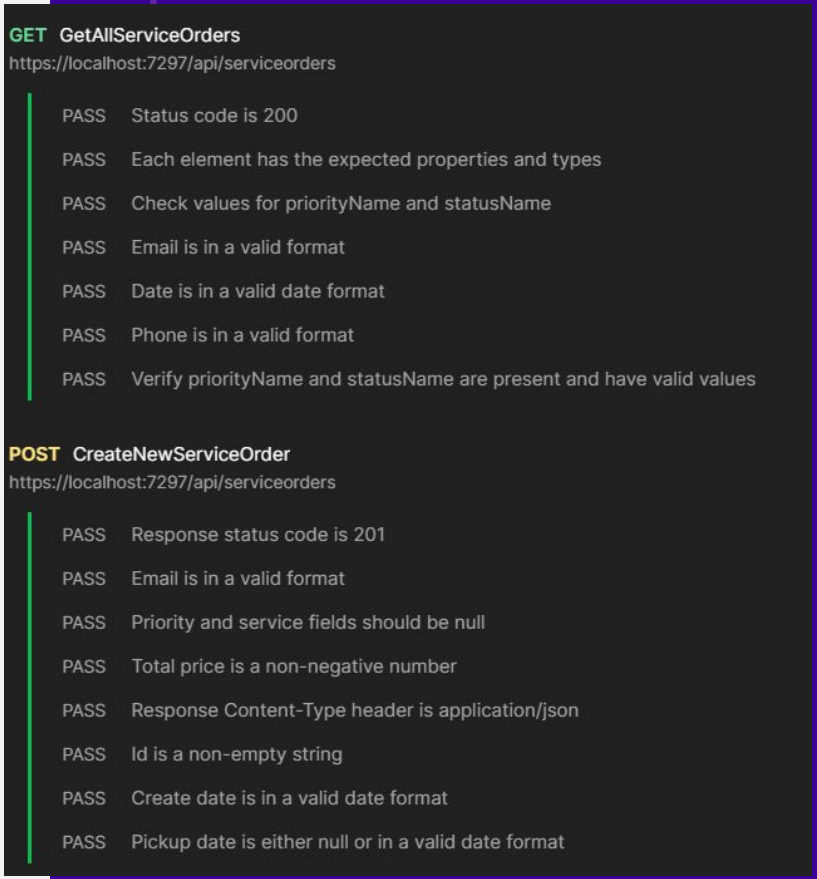
- Plugins
- API-Backend erstellen
- Skripte
  - Warum unfertig?
- Postman-Testprojekt

```
public MappingProfile()
{
    // Automate the mapping of DTOs to models and vice versa
    CreateMap<ServiceOrderModel, CreateServiceOrderDto>();
    CreateMap<CreateServiceOrderDto, ServiceOrderModel>();

    CreateMap<ServiceOrderModel, ServiceOrderDto>();
    CreateMap<ServiceOrderDto, ServiceOrderModel>();

    CreateMap<ServiceOrderModel, UpdateServiceOrderDto>();
    CreateMap<UpdateServiceOrderDto, ServiceOrderModel>();

    CreateMap<UserModel, UserDto>();
    CreateMap<UserDto, UserModel>();

    CreateMap<UserModel, CreateUserDto>();
    CreateMap<CreateUserDto, UserModel>();

    CreateMap<UserModel, LoginUserDto>();
    CreateMap<LoginUserDto, UserModel>();
```

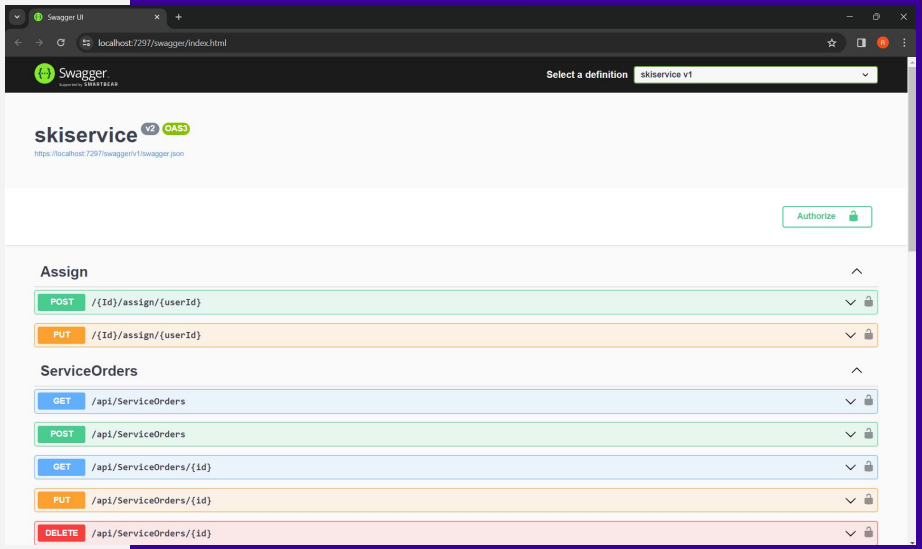
05

Kontrollieren

- Postman-Tests
- Swagger



06 Auswerten & Demo



07

Fazit

- Lessons Learned
  - Überraschungen
- Eigene Meinung
- Wie fand ich das Projekt

```
public MappingProfile()
{
    // Automate the mapping of DTOs to models and vice versa
    CreateMap<ServiceOrderModel, CreateServiceOrderDto>();
    CreateMap<CreateServiceOrderDto, ServiceOrderModel>();

    CreateMap<ServiceOrderModel, ServiceOrderDto>();
    CreateMap<ServiceOrderDto, ServiceOrderModel>();

    CreateMap<ServiceOrderModel, UpdateServiceOrderDto>();
    CreateMap<UpdateServiceOrderDto, ServiceOrderModel>();

    CreateMap<UserModel, UserDto>();
    CreateMap<UserDto, UserModel>();

    CreateMap<UserModel, CreateUserDto>();
    CreateMap<CreateUserDto, UserModel>();

    CreateMap<UserModel, LoginUserDto>();
    CreateMap<LoginUserDto, UserModel>();
```

Danke!

Noch Fragen?